FILE "slist" in LAF "slistext"

TS 1000 SCROLLIST

add → by Gregory C. Harder

SCROLLIST is a short machine code program which will make a useful addition to your utilities package. This routine produces a continuous listing of your BASIC program, which scrolls at a selectable speed. Since the scrolled listing is almost identical to the normal listing, it is very useful for program debugging, perusing a long program, or locating specific lines and commands within the BASIC program. I say "almost identical" because SCROLLIST will even display lines containing double CHR$ 118's that "hide" machine-code REMs. This means you can scan such machine-code REMs, to get an idea where the code is, where the data is, and how long it is.

The routine requires two parameters to be passed to it; these are the starting line number, and the scroll speed. In order to facilitate this, the calculator stack is used to temporarily hold the parameters. This requires that the routine be called in the following manner:

```
LET SLIST=LINE + (SPEED + USR CALL)
```

Where:

LINE = the line number at which to start scrolling.

SPEED = scrolling speed in the range of 1 to 255, with 1 being the fastest.

CALL = the USR address.

This method of passing parameters via the calculator stack is termed "inverted syntax," and is described in detail by D. Woods in his book "Machine Language Programming on the Timex/Sinclair 1500 and 1000".

The routine can be used as a direct call, or from within the BASIC program. When the routine is called it will display a listing of the BASIC program starting at the specified line number, at the selected scroll speed. To PAUSE the scrolling action press the M key. To CONTINUE press the C key. Pressing the A key will STOP the program and produce a normal LISTing of the BASIC program, starting at the last line number printed. The program will stop automatically, with report E, when the end of the BASIC program is reached.

LOAD "SCROLLIST" to get TS2040 listing of "LISTING 1"

ENTERING THE PROGRAM

To maximize the usefulness of SCROLLIST it should be located within the 8K to 16K area or above RAMTOP. If you put it above RAMTOP, be sure to lower RAMTOP before uploading, unless you modified your machine to run m/c above 32K.

Listing 1 is a combined hex loader and relocator, which contains the code for SCROLLIST. Enter listing 1, then run the program and follow the instructions. Be sure to make a note of the USR call for later reference. This will always be 33 bytes greater than your selected base address.

Note that the last character in lines 40, 70, 100, and 130 is a graphic space.

Listing 2 is an annotated assembly listing of the routine. You may notice that it uses numerous ROM calls to do most of the dirty work and keep it short (only 221 bytes). The ROM SCROLL routine is not used, because SCROLLIST needs a 24-line scroll.

Be sure to put the computer into SLOW mode before using SCROLLIST, or else you will not see the display.

LOAD "SLST-M/C"

```
ADDR HEXCODE  NAME MNEMONIC
============================
2000 2A0C40   SCRL LD HL,(DFIL)
2003 54           LD D,H
2004 5D           LD E,L
2005 012100       LD BC,0021
2008 09           ADD HL,BC
2009 01F702       LD BC,02F7
200C EDB0         LDIR
200E 2A1040       LD HL,(VARS)
2011 0620         LD B,20
2013 2B           DEC HL
2014 2B       CLER DEC HL
2015 71           LD (HL),C
2016 10FC         DJNZ CLER
2018 FD463C       LD B,(SPEE)
201B 0B       DLAY DEC BC
201C 78           LD A,B
201D B1           OR C
201E 20FB         JR NZ DLAY
2020 C9           RET
2021 CDCD15   SLST CALL FP2A
2024 323C40       LD (SPEE),A
2027 CD8A15       CALL FPBC
202A 60           LD H,B
202B 69           LD L,C
202C AF           XOR A
202D 322240       LD (DFSZ),A
2030 CDD809       CALL LNAD
2033 E5       MORE PUSH HL
2034 010016       LD BC,1600
2037 CDF508       CALL PRAT
203A E1           POP HL
203B E5           PUSH HL
203C CDBB05       CALL LNUM
203F ED530A40     LD (EPPC),DE
2043 E1           POP HL
2044 CDA50A       CALL PRLN
2047 23           INC HL
2048 4E           LD C,(HL)
2049 23           INC HL
204A 46           LD B,(HL)
204B 0B       NEXT DEC BC
204C 23           INC HL
204D D9       KEY? EXX
204E CDBB02       CALL KBSC
2051 44           LD B,H
2052 4D           LD C,L
2053 2C           INC L
2054 C4BD07       CALL NZ DCOD
2057 7E           LD A,(HL)
2058 D9           EXX
2059 FE32     KEYM CP 32
205B 2004         JR NZ KEYC
205D FDCB01E6     SET 4,(FLGS)
2061 FE28     KEYC CP 28
2063 2815         JR Z CONT
2065 FE26     KEYA CP 26
2067 200B         JR NZ BIT4
2069 CD2A0A       CALL CLS-
206C 2A0A40       LD HL,(EPPC)
206F CD3B07       CALL LIST
```

```
2072 1858           JR ER-E
2074 FDCB0166 BIT4  BIT 4,(FLGS)
2078 20D3           JR NZ KEY?
207A FDCB01A6 CONT  RES 4,(FLGS)
207E 78             LD A,B
207F B1             OR C
2080 2844           JR Z END?
2082 FDCB01AE       RES 5,(FLGS)
2086 3A3A40   SCR?  LD A,(ROW=)
2089 FE01           CP 01
208B 2011           JR NZ NUM?
208D D9             EXX
208E CD0020         CALL SCRL
2091 FD4639         LD B,(COL=)
2094 3E21           LD A,21
2096 90             SUB B
2097 4F             LD C,A
2098 0616           LD B,16
209A CDF508         CALL PRAT
209D D9             EXX
209E 7E       NUM?  LD A,(HL)
209F CDB407         CALL NO.?
20A2 2007           JR NZ TKN?
20A4 0B             DEC BC
20A5 0B             DEC BC
20A6 0B             DEC BC
20A7 0B             DEC BC
20A8 0B             DEC BC
20A9 18A0           JR NEXT
20AB CB77     TKN?  BIT 6,A
20AD 2814           JR Z PR-A
20AF FD5E3A         LD E,(ROW=)
20B2 C5             PUSH BC
20B3 CD4B09         CALL TOKN
20B6 C1             POP BC
20B7 3A3A40         LD A,(ROW=)
20BA BB             CP E
20BB 288E           JR Z NEXT
20BD FDCB01EE       SET 5,(FLGS)
20C1 1888           JR NEXT
20C3 D7       PR-A  RST 10H
20C4 1885           JR NEXT
20C6 23       END?  INC HL
20C7 7E             LD A,(HL)
20C8 FE76           CP 76
20CA 2002           JR NZ AGIN
20CC CF       ER-E  RST 08H
20CD 0D              ERROR E
20CE FDCB016E AGIN  BIT 5,(FLGS)
20D2 E5             PUSH HL
20D3 C40020         CALL NZ SCRL
20D6 CD0020         CALL SCRL
20D9 E1             POP HL
20DA C33320         JP MORE
```

File "slistanot" in LAF "slistex

```
>UP SCROLL< Locate D_FILE.
;Set destination address.

;Set source address.

;Set counter.
;Transfer.
;Locate VARS.
;Set counter.
;Decrease pointer.
;Clear last row.


;Get scroll speed.
;Scroll speed delay.



;Return, done scrolling.
;>START< Get the scroll speed.
;Store speed.
;Get starting line number.
;Transfer line number to HL.

;Clear reg. A.
;Set DF_SZ for full display
;Locate start addr. of line.
;Stack address pointer.
;Reg. BC=PRINT AT co-ords.
;Set print position.
;Restore address pointer.
;Stack again.
;Reg. DE=current line no.
;Set EPPC to line number.
;Restore address pointer.
;Print current line number.
;Reg. BC= length of line.



;Decrease the counter.
;Increase address pointer.
;Save main registers.
;Scan the keyboard.
;Set reg. BC for decode.

;Test for no key-press.
;If key-press, then decode.
;Set A to character code.
;Restore main registers.
;Last key = M?
;No, test for key C.
;Yes, set PAUSE flag.
;Last key = C?
;Yes, then continue.
;No, last key = A?
;No, then test PAUSE flag.
;Yes, clear screen.
;Set EPPC to last line value.
;LIST starting at last line.
```

```
;Precautionary stop by ER-E.
;Test PAUSE flag.
;If set, loop back to KEY?.
;Reset PAUSE flag.
;Line counter=zero?

;Yes, test for end of prog.
;No, reset overflow flag.
;Check S_POSN row value.
;Row = 1?
;No, test for a number code.
;Yes, save main registers.
;Scroll the line.
;Set next column position.



;Set next PRINT AT position.
;Restore main registers.
;A = current character code.
;Is it a floating-point no.?
;No, test for token code.
;Yes, decrease counter by 5.



;Do next character.
;Is character a token?
;No, then print the character.
;Yes, get current row value.
;Save counter.
;Expand and print the token.
;Restore the counter.
;Compare old and new row values.

;If zero, do next character.
;If not zero, set overflow flag.
;Do next character.
;Print character via RST 10h
;Do next character.
;Increase address pointer.
;Fetch the character.
;End of BASIC program?
;No, then continue.
;Yes, then stop with report E.
;
;Test the overflow flag.
;Save address pointer.
;If overflow, do an extra scroll.
;Scroll last line.
;Restore address pointer.
;Do next program line.
```

ORIGINAL

TS 1000 SCROLLIST

SCROLLIST is a short machine code program which will make a useful addition to your utilities package. Essentially, the program produces a listing of your basic program which scrolls at a selectable speed. Since the scrolled listing is identical to the normal listing, it is very useful for program debugging, perusing a long program, or locating specific lines and commands within the basic program.

The routine requires two parameters to be passed to it, these are the starting line number and the scroll speed. In order to facilitate this the calculator stack is used to temporarily hold the parameters. This requires that the routine be called in the following manner:

LET SLIST=LINE + (SPEED + USR CALL)

Where:

LINE = the line number at which to start scrolling.

SPEED = scrolling speed in the range of 1 to 255, with 1 being the fastest.

CALL = the USR address.

The method of passing parameters via the calculator stack is termed "inverted syntax" and is described in detail by D. Woods in his book "Machine Language Programming on the Timex/Sinclair 1500 and 1000".

The routine can be used as a direct call or from within the basic program.

When the routine is called it will display a listing of the basic program starting at the specified line number and selected scroll speed. To PAUSE the scrolling action press the P key. To CONTINUE press the C key. Pressing the A key will STOP the program and produce a normal LISTing of the basic program starting at the last line number printed. The program will stop automatically, with report E, when the end of the basic program is reached.

ENTERING THE PROGRAM

To maximize the usefulness of SCROLLIST it should be located within the 0K to 16K area or above RAMTOP.

Listing 1 is a combined hex loader and relocator, which contains the code for SCROLLIST. Enter listing 1 then run the program and follow the instructions. Be sure to make a note of the USR call for later reference.

Note that the last character in lines 40, 70, 100, and 130 is a graphic space.

Listing 2 is an annotated assembly listing of the routine. You may notice that it uses numerous ROM calls to do most of the dirty work and keep it short. The ROM SCROLL routine is not used because SCROLLIST needs a 24 line screen scroll.

Be sure to put the computer into SLOW mode before using SCROLLIST or else you will not see the display.

LISTING 1: SCROLLIST LOADER.

```
  10 REM ===== SCROLLIST =====
  20 REM
  30 FAST
  40 LET D$="2A 0C 40 54 5D 01 2
1 00 09 01 F7 02 ED B0 2A 10 40
06 20 2B 2B 71 10 FC FD 46 3C 0B
 78 B1 20 FB C9 CD CD 15 32 3C 4
0 CD 8A 15 60 69 AF 32 22 40 CD
D8 09■"
  50 REM
  60 REM
  70 LET D$=D$+"E5 01 00 16 CD F
5 08 E1 E5 CD BB 05 ED 53 0A 40
E1 CD A5 0A 23 4E 23 46 0B 23 D9
 CD BB 02 44 4D 2C C4 BD 07 7E D
9 FE 32 20 04 FD CB 01 E6 FE 28
28 15 FE 26 20 0B CD 2A 0A 2A 0A
 40 CD 3B 07 18 58■"
  80 REM
  90 REM
 100 LET D$=D$+"FD CB 01 66 20 D
3 FD CB 01 A6 78 B1 28 44 FD CB
01 AE 3A 3A 40 FE 01 20 11 D9 CD
 00 00 FD 46 39 3E 21 90 4F 06 1
6 CD F5 08 D9 7E CD B4 07 20 07
0B 0B 0B 0B 0B 18 A0■"
 110 REM
 120 REM
 130 LET D$=D$+"CB 77 28 14 FD 5
E 3A C5 CD 4B 09 C1 3A 3A 40 BB
28 8E FD CB 01 EE 18 88 D7 18 85
 23 7E FE 76 20 02 CF 0D FD CB 0
1 6E E5 C4 00 00 CD 00 00 E1 C3
00 00■"
 140 REM
 150 REM ===== CHECKSUM =====
 160 REM
 170 IF LEN D$<>663 THEN GOTO 50
0
 180 LET SUM=0
 190 FOR N=1 TO 663
 200 LET SUM=SUM+CODE D$(N)
 210 NEXT N
 220 IF SUM<>15796 THEN GOTO 500
 230 REM ===== LOADER =====
 240 PRINT "WHERE DO YOU WANT TO
 LOCATE     SCROLLIST?";AT 4,0;"
YOU WILL NEED AT LEAST 221 FREE
BYTES."
 250 INPUT ADDR
 260 LET A=ADDR
 270 FOR N=1 TO 663 STEP 3
 280 POKE A,16*CODE D$(N)+CODE D
$(N+1)-476
 290 LET A=A+1
 300 NEXT N
 310 REM ===== CALLS/JUMPS =====
 320 LET LO=16434
 330 LET HI=16435
 340 RAND ADDR
 350 POKE ADDR+143,PEEK LO
 360 POKE ADDR+144,PEEK HI
 370 POKE ADDR+212,PEEK LO
 380 POKE ADDR+213,PEEK HI
 390 POKE ADDR+215,PEEK LO
 400 POKE ADDR+216,PEEK HI
 410 RAND ADDR+51
 420 POKE ADDR+219,PEEK LO
 430 POKE ADDR+220,PEEK HI
 440 LET CALL=ADDR+33
 450 CLS
```

```
 460 PRINT "ROUTINE TRANSFERED."
,,,"TO USE SCROLLIST:",,,"LET CA
LL=LINE+(SPEED+USR ";CALL;")"
 470 PRINT ,,,,"WHERE:",,,,"LINE
  = STARTING LINE NUMBER",,,"SPE
ED = SCROLL SPEED (1 TO 255)"
 480 PRINT AT 16,0;"BE SURE TO M
AKE A NOTE OF THE   USR ADDRESS
FOR LATER REFERENCE."
 490 STOP
 500 PRINT "D$ ERROR.";AT 2,0;"C
HECK LINES 40, 70, 100, AND 130"
 510 STOP
 520 SAVE "SCROLLIS▒"
 530 RUN
```

LISTING 2: SCROLLIST PROGRAM.

```
ADDR HEXCODE   NAME MNEMONIC
============================
2000 2A0C40    SCRL LD HL,(DFIL)   >UP SCROLL< Locate D-FILE.
2003 54             LD D,H         ;Set destination address.
2004 5D             LD E,L
2005 012100         LD BC,0021     ;Set source address.
2008 09             ADD HL,BC
2009 01F702         LD BC,02F7     ;Set counter.
200C EDB0           LDIR           ;Transfer.
200E 2A1040         LD HL,(VARS)   ;Locate VARS.
2011 0620           LD B,20        ;Set counter.
2013 2B             DEC HL         ;Decrease pointer.
2014 2B        CLER DEC HL         ;Clear last row.
2015 71             LD (HL),C
2016 10FC           DJNZ CLER
2018 FD463C         LD B,(SPEE)    ;Get scroll speed.
201B 0B        DLAY DEC BC         ;Scroll speed delay.
201C 78             LD A,B
201D B1             OR C
201E 20FB           JR NZ DLAY
2020 C9             RET            ;Return, done scrolling.
2021 CDCD15    SLST CALL FP2A      >START< Get the scroll speed.
2024 323C40         LD (SPEE),A    ;Store speed.
2027 CD8A15         CALL FPBC      ;Get starting line number.
202A 60             LD H,B         ;Transfer line number to reg. HL
202B 69             LD L,C
202C AF             XOR A          ;Clear reg. A.
202D 322240         LD (DFSZ),A    ;Set DF-SZ for full size display
2030 CDD809         CALL LNAD      ;Locate starting addr. of line.
2033 E5        MORE PUSH HL        ;Stack address pointer.
2034 010016         LD BC,1600     ;Reg. BC = PRINT AT coordinates.
2037 CDF508         CALL PRAT      ;Set print position.
203A E1             POP HL         ;Restore address pointer.
203B E5             PUSH HL        ;Stack again.
203C CDBB05         CALL LNUM      ;Reg. DE = current line number.
203F ED530A40       LD (EPPC),DE   ;Set EPPC to line number.
2043 E1             POP HL         ;Restore address pointer.
2044 CDA50A         CALL PRLN      ;Print the current line number.
2047 23             INC HL         ;Reg. BC = length of line.
2048 4E             LD C,(HL)
2049 23             INC HL
204A 46             LD B,(HL)
204B 0B        NEXT DEC BC         ;Decrease the counter.
204C 23             INC HL         ;Increase address pointer.
204D D9        KEY? EXX            ;Save main registers.
204E CDBB02         CALL KBSC      ;Scan the keyboard.
2051 44             LD B,H         ;Set reg. BC for decode.
2052 4D             LD C,L
2053 2C             INC L          ;Test for no key press.
2054 C4BD07         CALL NZ DCOD   ;If key press, then decode.
2057 7E             LD A,(HL)      ;Set reg. A to character code.
2058 D9             EXX            ;Restore main registers.
2059 FE32      KEYM CP 32          ;Last key = M?
205B 2004           JR NZ KEYC     ;No, test for key C.
205D FDCB01E6       SET 4,(FLGS)   ;Yes, then set "pause" flag.
2061 FE28      KEYC CP 28          ;Last key = C?
2063 2815           JR Z CONT      ;Yes, then continue.
2065 FE26      KEYA CP 26          ;No, last key = A?
2067 200B           JR NZ BIT4     ;No, then test "pause" flag.
2069 CD2A0A         CALL CLS-      ;Yes, clear screen.
206C 2A0A40         LD HL,(EPPC)   ;Set EPPC to last line value.
206F CD3B07         CALL LIST      ;LIST starting at last line.
2072 1858           JR ER-E        ;Precautionary stop by ER-E.
2074 FDCB0166  BIT4 BIT 4,(FLGS)   ;Test "pause" flag.
2078 20D3           JR NZ KEY?     ;If set, then loop back to KEY?.
207A FDCB01A6  CONT RES 4,(FLGS)   ;Reset "pause" flag.
207E 78             LD A,B         ;Line counter = zero?
207F B1             OR C
2080 2844           JR Z END?      ;Yes, then test for end of prog.
2082 FDCB01AE       RES 5,(FLGS)   ;No, then reset overflow flag.
```

```
2086 3A3A40    SCR? LD A,(ROW=)   ;Check S-POSN row value.
2089 FE01           CP 01         ;Row = 1?
208B 2011           JR NZ NUM?    ;No, then test for a number code
208D D9             EXX           ;Yes, then save main registers.
208E CD0020         CALL SCRL     ;Scroll the line.
2091 FD4639         LD B,(COL=)   ;Set next column print position.
2094 3E21           LD A,21
2096 90             SUB B
2097 4F             LD C,A
2098 0616           LD B,16
209A CDF508         CALL PRAT     ;Set next PRINT AT position.
209D D9             EXX           ;Restore main registers.
209E 7E        NUM? LD A,(HL)     ;Reg. A = current character code
209F CDB407         CALL NO.?     ;Is it a floating point number?
20A2 2007           JR NZ TKN?    ;No, then test for a TOKEN code.
20A4 0B             DEC BC        ;Yes, then decrease counter by 5
20A5 0B             DEC BC
20A6 0B             DEC BC
20A7 0B             DEC BC
20A8 0B             DEC BC
20A9 18A0           JR NEXT       ;Do next character.
20AB CB77      TKN? BIT 6,A       ;Is character a TOKEN?
20AD 2814           JR Z PR-A     ;No, then print the character.
20AF FD5E3A         LD E,(ROW=)   ;Yes, get current row value.
20B2 C5             PUSH BC       ;Save counter.
20B3 CD4B09         CALL TOKN     ;Expand and print the TOKEN chr.
20B6 C1             POP BC        ;Restore the counter.
20B7 3A3A40         LD A,(ROW=)   ;Compare old and new row values.
20BA BB             CP E
20BB 288E           JR Z NEXT     ;If zero, do next character.
20BD FDCB01EE       SET 5,(FLGS)  ;If not zero, set overflow flag.
20C1 1888           JR NEXT       ;Do next character.
20C3 D7       PR-A  RST 10H       ;Print character via RST 10h.
20C4 1885           JR NEXT       ;Do next character.
20C6 23       END?  INC HL        ;Increase address pointer.
20C7 7E             LD A,(HL)     ;Fetch the character.
20C8 FE76           CP 76         ;End of basic program?
20CA 2002           JR NZ AGIN    ;No, then continue.
20CC CF       ER-E  RST 08H       ;Yes, then stop with report E.
20CD 0D              ERROR E      ;
20CE FDCB016E AGIN  BIT 5,(FLGS)  ;Test the overflow flag.
20D2 E5             PUSH HL       ;Save address pointer.
20D3 C40020         CALL NZ SCRL  ;If overflow, do an extra scroll
20D6 CD0020         CALL SCRL     ;Scroll last line.
20D9 E1             POP HL        ;Restore address pointer.
20DA C33320         JP MORE       ;Do next program line.
```